Kristianstad University
www.hkr.se

Fredrik Frisk
Computer Science Department

Content and purpose of this lab

The main focus of the lab is to do classification.

Save the code you are writing in this lab for future use. To pass the lab you need to solve/program the different bullet points and be able to explain your results. If you are not finished with the all the bullet points the remaining ones are a part of the required preparations for part 3 of the labs.

Required Preparations

- You need to finish part one of the lab and be able to show the result in the beginning of the lab.

You need to do additional recordings to the recordings you did in part 1. For details as sampling time, number of recordings per class, how long the recording you should do etc, look in the chapter “Acquire some data” from part 1 of the lab. The following classes should be recorded.

- Sitting
- Running
- Jumping
- Any other movement of your choice which is not walking, running or jumping.

You need to be able to show a python function that can take any number of pickle-files belonging to the same class and combine them to one dataframe adding the class as a seventh column.

Input:

- A string with the name of the train folder or the test folder.
- A string with the name of the class

Output:

- A dataframe containing the created dataframe described above

The function should also store the created dataframe as a pickle file.

Be prepared to show this function and the output from this function, that is the dataframe with 7 columns, for the recorded classes you have stored as a preparation for this lab.

Other Preparations (These will save you time)

Work through the two exercises you can find on Canvas the previous week, both the KNN exercise and the decision tree exercise.

Classification of different positions

Tip: Create one notebook for KNN and another one for the Decision Tree.

We will use the K-nearest neighbour algorithm and decision trees in this lab. In this part you should only work with the stationary positions:

- Standing
- Sitting
- Laying down

We start with working with the KNN algorithm. You should use crossvalidation for evaluating the model.

- Make a choice of the the number of subsets that you use for your crossvalidation. Motivate the choice

For the gridsearch use GridSearchCV which you find in the previous weeks´ exercise. The hyperparameter you will use for the gridsearch is k, that is the number of neighbours.

- Make a choice of the range of k-values you will use. Motivate the choice.
- Train the model.
- What is the optimal k-value?
- What is the accuracy?
- Plot the accuracy as a function of the k-value. You need to extract that information from the model. How sensitive is the performance of the model for different k-values?

Now it is time to look at the test set with the optimal k-value

- Use the model on the test set and acquire both the accuracy and the plot the confusion matrix.
- Explain the difference between the accuracy for the validation set and the test set
- Explain the result in the confusion matrix

Now we continue with the decision tree. You should use the decision tree classifier you find in scikit-learn.

- Use the gridsearchCV as you did for KNN. As hyperparameters use max_depth and criterion. Use accuracy as previously.
- Does the criterion have an impact on the accuracy? Make an investigation so you can answer this question
- Plot the tree with the optimal hyperparameters

Now it is time you look at the test set.

- Using the treemodel above to predict the accuracy.
- Plot the confusion matrix as well
- Explain the difference between the accuracy for the validation set and the test set
- Explain the result you see in the confusion matrix

As a last step it is time to compare the results from the KNN-model and the decision tree.

- What is the difference between the acurracies?
- What is the difference between the confusion matrices?

Keep the note from this part of the lab. This type of analysis is what you need for the lab report and any type of ML-analysis.

Kristianstad University Sweden

Normalisation

Start a new notebook for this part.

We will work with the minmax scaler in the Scikit-learn package. Before scale the data we will look at some statistics. Below we will work with the training files only.

- Use the describe method for all seven classes. What are the differences between all classes? What statistic measure is different for the classes? Make boxplots of each class as well.
- Use one of the outlier detection methods presented in the course. Which one did you choose and why? How many outliers did you detect? In which classes did you detect them?
- If you found any outliers, remove them by replacing then with the mean value of the neighboring datapoints/instances. Store the new dataframe in a new pickle file. These new pickle files we will work with during the remainder of the labs.

Time for normalisation, **use a new notebook here**

- Is it needed using normalisation for the decision tree classifier? Why or why not?
- Is it needed using normalisation for the KNN classifier? Why or why not?

Independent of you answer below we will evaluate the minmax-scaler for the KNN algorithm. You are free to use a pipeline for organising the steps.

- Normalise the features so they all in the range from -1 to 1.
- Work through everything you did before when you evaluated the performance using the KNN-algorithm (classification of different position chapter)
- Compare the results with the previous analysis.
- Which one give the best result, KNN, KNN-normalised or Decision tree?